REMARKS

With this response, claims 1, 3-16, 18-25, 27-33, and 37-41 are pending in the present application.

The Applicants has carefully and thoughtfully considered the Office Action and the comments therein. For the reasons given below, it is submitted that this application is in condition for allowance. Thus, continued examination of the present application as amended is hereby requested.

Omissions, Errors, Inconsistencies, and/or Ambiguities in Office Action

As discussed below, the Office Action contains certain omissions, errors, inconsistencies, and/or ambiguities which make it difficult for Applicants to respond effectively and precisely. Applicants respectfully requests that these be addressed if another Office Action is issued.

Rejections Under 35 USC § 103(a) – Three-way Combination of Coad, Little, and Liu

On pages 2-20, the Office Action rejects claims 1, 3-5, 7-16, 18-20, 22-25, 27-33, and 37-41 as being unpatentable over U.S. Patent No. 6,851,105 to Coad et al. (Coad) in view of U.S. Patent No. 7,047,518 to Little et al. (Little) and in further view of U.S. Patent No. 7,296,256 to Liu et al. (Liu). Applicants respectfully traverse the rejection. Coad, Little, and Liu are discussed first, followed by the Office Action's rejections of claims 1, 3-5, 7-16, 18-20, 22-25, 27-33, and 37-41.

A. Coad

Coad discloses "methods and systems for generating, applying, and defining patterns for existing code in order to improve the design and efficiency of the existing code." Coad, col. 1, l. 28-30 and col. 6, ll. 42-47. Patterns for software development ensure sound code architecture, help resolve common, recurring software development problems, and allow developers to organize, document, and produce more maintainable software. Coad, col. 6, l. 58-67.

Methods and systems disclosed by Coad utilize a software development tool that generates a pattern instance and applies the pattern instance to a portion of existing code to improve the design of the existing code. Coad, col. 6, ll. 42-47. The software development tool can also be used to create a new pattern from existing code. Coad, col. 6, ll. 42-47.

When improving the design of existing code, the software development tool performs three steps. First, the software development tool receives an feature type to distinguish the type of pattern that is to be created. Coad, col. 8, l. 51-53 and Figure 2, step 202. Second, the software development tool displays a list of pattern options that are applicable to the selected feature type and from which the developer can choose. Coad, col. 10, l. 1-3 and Figure 2, step 204. Third, once the developer has selected a pattern, the software generation tool applies the pattern to the code. Coad, col. 24, l. 64-67 and FIG. 2, step 222.

In addition, the creation of patterns in existing source code may be depicted in a diagram. See, e.g., Coad, Figs. 9-11, diagrams 9022 and 1022; and Coad, col. 25, ll. 36-46.

B. Little

Little discloses "a system for integrated computer software application development and modeling." Little, Abstract. The system provides tools to assist a software engineer in the refinement of a model of a software application by folding existing application and database knowledge into the model of the software application. Little, col. 1, l. 64-67 and col. 9, l. 41-53. Particularly, "Design Patterns and Model Verification tools assist the software developer by performing much of the routine work necessary to complete a model [of a software application]." Little, col. 9, l. 53-56. For example, tools may apply a design pattern to the user's model of the software application in order to improve the software model's integration and efficiency. Little, col. 10 37-40. Once the software model is complete, "source code generation creates the implementation from the [software] model and helps keep the model and source code synchronized." Little, col. 9, l. 56-58

C. Liu

Liu discloses combining "inference techniques with queuing models to automate the process of performance modeling and optimization of IT systems[, such as the e-business service structure depicted in FIG. 4,] and applications." Liu, col. 2, l. 36-39. Liu "provides an end-to-end self-tuning, and flexible method that allows the building of performance models [] based on the system monitoring information." Liu, col. 4, l. 14-17. The model may then "provide insights of the capabilities of a Website, as well as a better understanding of the trade-offs between scalability, [Quality of Service (QoS)], capacity cost, operations risk, etc." and, therefore, plan and develop better IT solutions. Liu, col. 4, l. 20-29.

Liu builds a network model, such as the one depicted in FIG. 5, by obtaining the parameters of the service demands for each class at each generic server in the network model. Liu, col. 64-67. These server time parameters may be inferred from the most common measurements, such as the system throughput, utilization of the servers, and end-to-end response times. Liu, col. 8, l. 14-17. The methodology to infer these service parameters includes two aspects. Liu, col. 8, l. 25-26. First, the parameters can be inferred "if the underlying network has closed-form performance expressions (or good approximation[])." Liu, col. 8, l. 26-32. The use of close-form equations is depicted in col. 8, l. 34-57 and col. 9, l. 5-27 of Liu. Second, "if closed-form expressions or analytic approximations are not available, the present invention may rely on discrete-event simulator 611 together with a set of meta-heuristic search methods 613 including various tabu search and/or simulated annealing algorithms to search for the optimal set of parameters." Liu, col. 9, l. 37-42. The model may be build by, for example, module 603 in FIG. 6.

"Once a valid performance model has been established, the above closed-form expressions or the recursive mean-value analysis algorithm can be used to predict performance, optimize existing IT infrastructure, and to suggest cost-effective architecture design through deployment and operations. Those functions may be implemented in module 605 in FIG. 6." Liu, col. 10, l. 30-36.

D. Rejections of claims 1, 3-5, 7-16, 18-20, 22-25 and 27-40.

Regarding claim 1, Applicants initially note that **the Office Action addresses at least one feature that does not appear in claim 1** – namely "each embedded code relating to a characteristic

of code to be generated from the graphical interface" – and is, therefore, not addressed in this response. Applicants respectfully request that this feature be deleted from any future Office Actions.

Further, Coad, Little, and Liu taken either singly or in any reasonable combination, do not disclose or render obvious the claimed invention for at least the following five reasons.

1. The Office Action does not address "prompting a user to specify at least one code generation goal from a plurality of code generation goals," as recited in claim 1.

First, **the Office Action does not address "prompting a user to specify at least one code generation goal from a plurality of code generation goals," as recited in claim 1**. In particular, the Office Action relies on Coad to teach every feature of claim 1 except for "generating code for a code generation goal," "the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment," "changing parameters of the graphical model that are inconsistent with the at least one code generation goal," "the graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equations," and "the graphical model being capable of simulation based on the equations." Applicants agree that Coad does not teach these features. However, in addition to not teaching these features of claim 1, Coad fails to disclose "prompting a user to specify at least one code generation goal from a plurality of code generation goals."

In rejecting claim 1, the Office Action discusses a limitation of "prompting a user to specify the embedded code from a plurality of embedded code" and states that it corresponds to "display[ing] pattern options corresponding to selected feature type" in Figure 2 of Coad. Office Action, pg. 3. However, claim 1 does not recite "prompting a user to specify the embedded code from a plurality of embedded code." For the purposes of this response, the Office Action is assumed to have aligned the recited "prompting a user to specify at least one code generation goal from a plurality of code generation goals" of claim 1 with "display pattern options corresponding to selected feature type" in Figure 2 of Coad.

Coad does not disclose "prompting a user to specify at least one code generation goal from a plurality of code generation goals," as recited in claim 1. Instead Coad discloses a "software development tool that generates [a] pattern instance and applies the pattern to a portion of existing code in order to improve the design of the existing code." Coad, col. 6, ll. 42-47. According to Coad, the software development tool may first receive an feature type to distinguish the type of pattern that is to be created. Coad, col. 7, l. 51-53 and Figure 2, step 202. The software development tool may then display a list of pattern options that are applicable to the selected feature type and from which the developer can choose. Coad, col. 10, l. 1-3 and Figure 2, step 204. The software generation tool may then apply the pattern to the code. Coad, col. 24, l. 64-67 and FIG. 2, step 222. **That is, Coad discloses applying a pattern to preexisting code, which is not equivalent to the "code generation goal" of claim 1** that "relat[es] to a characteristic of the code to be generated from the graphical model." Therefore, allowing a user to select a pattern to apply to existing code is not equivalent to "prompting a user to specify at least one code generation goal from a plurality of code generation goals," as recited in claim 1. Therefore, Coad does not disclose "prompting a user to specify at least one code generation goal from a plurality of code generation goals" as recited in claim 1.

Little does not overcome the failings of Coad. Instead, Little discloses a system that provides tools to assist a software engineer in the refinement of a model by folding existing applications and database knowledge into the model. Little, col. 9, l. 41-53. The tools of Little may apply a design pattern to the user's model in order to improve the model's integration and efficiency. Little, col. 10 37-40. Once the model is complete, "source code generation creates the implementation from the model and helps keep the model and source code synchronized." Little, col. 9, l. 56-58. That is, Little mentions code generation in general terms, but does not teach a "code generation goal" that "relat[es] to a characteristic of the code to be generated from the graphical model." **Applying a design pattern to a user's model, as Little discloses, is not equivalent to "prompting a user to specify at least one code generation goal from a plurality of code generation goals," as recited in claim 1**. Hence, the combination of Coad and Little fail to teach or render obvious "prompting a user to specify at least one code generation goal from a plurality of code generation goals," as recited in claim 1.

Liu does not overcome the failings of Coad and Little. Instead, Liu discloses the creation of performance models of IT systems so that those systems can be analyzed and refined. Liu, col. 4, l. 20-29. Liu builds a network model, such as the one depicted in FIG. 5, by obtaining the parameters of the service demands for each class at each generic server in the network model. Liu, col. 64-67. The creation of the performance models in Liu is not equivalent to the "code generation goal" of claim 1 which "relat[es] to a characteristic of the code to be generated from the graphical model." **Therefore, creating a performance model, as Liu discloses, is not equivalent to "prompting a user to specify at least one code generation goal from a plurality of code generation goals," as recited in claim 1**. Hence, the combination of Coad, Little, and Liu fail to teach or render obvious "prompting a user to specify at least one code generation goal from a plurality of code generation goals," as recited in claim 1.

Furthermore, Applicants request that the recited "prompting a user to specify at least one code generation goal from a plurality of code generation goals" feature of claim 1 be properly and precisely addressed in the next Office Action, if one is issued.

2. References do not disclose or suggest "the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment," as recited in claim 1.

Second, neither Coad, Little, and Liu taken alone or in combination, disclose or suggest "the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment," as recited in claim 1. In rejecting claim 1, the Office Action states that Coad does not teach the claim feature quoted above. Applicants agree. In order to overcome the failings of Coad, the Office Action relies on the combination of Coad in view of Little. Applicants disagree with this combination.

In rejecting claim 1, the Office Action aligns the "code generation goal" feature of claim 1 with the "display pattern options corresponding to selected feature type" of Coad. Office Action, pg. 3. In addition, the Office Action aligns the "code generation goal" feature of claim 1 with application model of Little. Office Action, pg. 3. The Action also aligns the "being used to

generate embedded code from the graphical model in a graphical modeling environment" feature of claim 1 with the "new model feature option 164," the "import model feature option 166," the apply design pattern option 168," the "preparing for implementation option 170," "the validate model option 172," the "rose expert generate option 174" and the "edit implementation feature 176" of Little, and cites Little, col. 10, l. 25-55. Office Action, pg. 2.

Assuming, *arguendo*, that the "display pattern options corresponding to selected feature type" of Coad and the application model of Little may be considered "code generation goals," Coad and Little cannot be combined to teach "the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment," as asserted by the Office Action. As discussed above, Little discloses a system for the creation and refinement of a software model. Little, col. 1, l. 64-67 and col. 9, l. 41-53. Once the software model is complete, "source code generation creates the implementation from the [software] model." Little, col. 9, l. 56-58. Coad, however, discloses a "method[] and system[] for generating, applying, and defining patterns for existing code in order to improve the design and efficiency of the existing code." Coad, col. 1, l. 28-30 and col. 6, ll. 42-47. Thus, if Coad were combined with Little, a person of ordinary skill in the art would combine **the method and system of Coad, which defines patterns for existing code, with the source code generated by Little, and not to the application model of Little**. Therefore, Coad in view of Little does not disclose or suggest "the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment," as recited in claim 1.

Furthermore, Liu fails to overcome the deficiencies of both Coad and Little. Liu discloses the creation of performance models of IT systems so that those systems can be analyzed and refined. Liu, col. 4, l. 20-29. Liu builds a network model, such as the one depicted in FIG. 5, by obtaining the parameters of the service demands for each class at each generic server in the network model. Liu, col. 64-67. The creation of the performance models in Liu is not equivalent to "the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment," as recited in claim 1. In fact, Liu does not involve the creation of embedded code whatsoever. The combination of Coad, Little, and Liu, therefore, fails to teach or

render obvious "the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment," as recited in claim 1.

3. References do not disclose or suggest "changing parameters of the graphical model that are inconsistent with the at least one code generation goal," are recited in claim 1.

Third, neither Coad, Little, or Liu, taken alone or in combination, disclose or suggest "changing parameters of the graphical model that are inconsistent with the at least one code generation goal," are recited in claim 1. In rejecting claim 1, the Office Action states that Coad does not teach the claim feature quoted above. Applicants agree. In order to overcome the failings of Coad, the Office Action relies on the combination of Coad in view of Little. Applicants disagree with this combination.

In rejecting claim 1, the Office Action aligns the "code generation goal" feature of claim 1 with the "display pattern options corresponding to selected feature type" of Coad. Office Action, pg. 3. In addition, the Office Action aligns the "code generation goal" feature of claim 1 with the application model of and the Main Use Case diagram of Little. Office Action, pg. 3. The Action also aligns the "changing parameters of the graphical model that are inconsistent with" feature of claim 1 with the "the class diagrams and performance analysis [] done based on the Main Use Diagram" in col. 13, l. 40-55 of Little. Office Action, pg. 3.

Assuming, *arguendo*, that the "display pattern options corresponding to selected feature type" of Coad as well as the application model and/or the Main Use Case diagram of Little may be considered "code generation goals," Coad and Little cannot be combined to teach "changing parameters of the graphical model that are inconsistent with the at least one code generation goal," as asserted by the Office Action. As discussed above, Little discloses a system for the creation and refinement of a software model. Little, col. 1, l. 64-67 and col. 9, l. 41-53. Once the software model is complete, "**source code generation creates the implementation from the [software] model**." Little, col. 9, l. 56-58. Coad, on the other hand, discloses a "method[] and system[] for generating, applying, and defining patterns **for existing code** in order to improve the design and efficiency of the existing code." Coad, col. 1, l. 28-30 and col. 6, ll. 42-47. Thus, if Coad were

combined with Little, a person of ordinary skill in the art would combine **<u>the method and system of Coad, which defines patterns for existing code, with the source code generated by Little, and not to the application model of Little</u>**. Therefore, Coad in view of Little does not disclose or suggest "changing parameters of the graphical model that are inconsistent with the at least one code generation goal," as recited in claim 1.

Liu fails to overcome the deficiencies of both Coad and Little. Liu discloses the creation of performance models of IT systems so that those systems can be analyzed and refined. Liu, col. 4, l. 20-29. Liu builds a network model, such as the one depicted in FIG. 5, by obtaining the parameters of the service demands for each class at each generic server in the network model. Liu, col. 64-67. The creation of the performance models in Liu is not equivalent to "changing parameters of the graphical model that are inconsistent with the at least one code generation goal," as recited in claim 1. In fact, Liu does not involve the generation of code whatsoever. The combination of Coad, Little, and Liu, therefore, fails to teach or render obvious "the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment," as recited in claim 1.

<u>4. The Office Action does not address "generating embedded code in accordance with the at least one code generation goal," as recited in claim 1.</u>

Fourth, the Office Action does not address "generating embedded code in accordance with the at least one code generation goal," as recited in claim 1. Instead, the Office Action states that Coad fails to disclose "generating code for a code generation goal," Applicants agree. The Action further aligns "generating code for a code generation goal," with the "application model" of Little. Little, col. 10, l. 25-55. As noted above, claim 1 does not recite "generating code for a code generation goal." However, for the purposes of this response, the Office Action is assumed to have aligned the recited "generating embedded code in accordance with the at least one code generation goal" with the "application model" of Little.

Little does not disclose "generating embedded code in accordance with the at least one code generation goal," as recited in claim 1. Instead, Little discloses a system that provides tools to assist

a software engineer in the refinement of a model by folding existing applications and database knowledge into the model. Little, col. 9, l. 41-53. The tools of Little may apply a design pattern to the user's model in order to improve the model's integration and efficiency. Little, col. 10 37-40. Once the model is complete, "source code generation creates the implementation from the model and helps keep the model and source code synchronized." Little, col. 9, l. 56-58 (emphasis added). Thus, **Little discloses applying a design pattern to a user's model**, as opposed to "generating embedded code in accordance with the at least one code generation goal," as recited in claim 1. Therefore, Little does not disclose or suggest "generating embedded code in accordance with the at least one code generation goal," as recited in claim 1.

Liu does not overcome the failings of Little. Instead, Liu discloses the creation of performance models of IT systems so that those systems can be analyzed and refined. Liu, col. 4, l. 20-29. Liu builds a network model, such as the one depicted in FIG. 5, by obtaining the parameters of the service demands for each class at each generic server in the network model. Liu, col. 64-67. The creation of the performance models in Liu is not equivalent to "generating embedded code in accordance with the at least one code generation goal," as recited in claim 1. In fact, Liu does not involve the generation of embedded code whatsoever. The combination of Coad, Little, and Liu, therefore, fails to teach or render obvious "generating embedded code in accordance with the at least one code generation goal," as recited in claim 1.

Furthermore, Applicants request that the recited "generating embedded code in accordance with the at least one code generation goal," feature of claim 1 be properly and precisely addressed in the next Office Action, if one is issued.

5. The References cannot be combined to teach or suggest "the graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equations, the graphical model being capable of simulation based on the equations," as recited in claim 1.

Fifth, Coad, Little, and Liu cannot be combined to teach or suggest "the graphical model representing a dynamic system having time-changing behavior modeled with differential,

difference, and/or algebraic equations, the graphical model being capable of simulation based on the equations," as recited in claim 1. In rejecting claim 1, the Office Action states that neither Coad nor Little teaches the claim feature quoted above. Applicants agree. In order to overcome the failings of Coad and Little, the Office Action relies on the combination of Coad in view of Little and Liu. Applicants respectfully disagree with this combination.

In rejecting claim 1, the Office Action aligns the "graphical model" with various features of Coad, Little, and Liu. The Action also aligns the "graphical model" of claim 1 with the software patterns of Coad, the application model as well as the class diagrams of Little (Little, col. 10, l. 25-55 and col.13, l. 40-55), and the process of constructing a performance model of Liu (Liu, col. 9, l. 35 – col. 10, l. 57). Office Action, pgs. 3-4.

Assuming, *arguendo*, that the FIGS. 9-12 of Coad, the application model as well as the class diagrams of Little, and the process of constructing a performance model of Liu may all be considered "graphical models," Coad, Little, and Liu cannot be combined to teach "the graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equations, the graphical model being capable of simulation based on the equations," as asserted by the Office Action. As discussed above, Coad discloses a "method[] and system[] for generating, applying, and defining patterns for existing code in order to improve the design and efficiency of the existing code." Coad, col. 1, l. 28-30 and col. 6, ll. 42-47. Little discloses a system for the creation and refinement of a software model. Little, col. 1, l. 64-67 and col. 9, l. 41-53. Once the software model is complete, "source code generation creates the implementation from the [software] model." Little, col. 9, l. 56-58. Liu discloses the creation of performance models of IT systems so that those systems can be analyzed and refined. Liu, col. 4, l. 20-29.

Thus, if the "graphical model" of Coad, which relates to a software pattern, or the "graphical model" of Little, which relates to a software model, were replaced with the "graphical model" of Liu, the result would be an inoperable combination as neither Coad nor Little are able to handle the a "graphic model" of an entire IT system, as is recited in Liu. For example, a person of ordinary skill in the art would not be able to modify Coad, which discloses the application of patterns to existing code, and Little, which deals with modeling and generating source code for a software

application, so that they would be compatible with Liu, which models entire IT systems, such as an e-business architecture. Liu, col.7, l. 45-62 and FIGS. 4 and 5. Coad in view of Little and Liu, therefore, cannot be combined to disclose "the graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equations, the graphical model being capable of simulation based on the equations," as recited in claim 1.

Therefore, for the above five reasons, Coad, Little, and Liu, taken either singly or in any reasonable combination, fail to disclose or render obvious claim 1. Applicants believe that claim 1 is allowable and respectfully request that the above rejection of claim 1 be withdrawn and that claim 1 be allowed.

Dependent claims 2-13, 40, and 41 depend on claim 1 and are believed to be allowable for at least the same reasons as above. Therefore, Applicants respectfully request that the above rejection of claims 2-13 and 40 be withdrawn and that claims 2-13 and 40 be allowed.

E. Claims 14, 16, and 30-33

Independent claims 14, 16, and 30-33 recite subject matter similar to that recited in claim 1, which Applicants believe is allowable over Coad in view of Little and Liu. Therefore, Applicants believe claims 14, 16, and 30-33 are allowable for at least the reasons set forth above. Applicants respectfully request that the above rejection of claims 14, 16, and 30-33 be withdrawn and that claims 14, 16, and 30-33 be allowed.

Claim 14 recites "graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equations," which is similar to "graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equations" as recited in claim 1. Claim 14 recites "the graphical model being capable of simulation based on the equations," which is similar to "the graphical model being capable of simulation based on the equations" as recited in claim 1. Claim 14 recites "the code generation goal being used to generate embedded code from the graphical model, the code generation goal relating to a characteristic of the code to be generated from the

graphical model," which is similar to "the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment, the at least one code generation goal relating to a characteristic of the code to be generated from the graphical model" as recited in claim 1. Claim 14 recites "generating embedded code in accordance with the code generation goal," which is similar to "generating embedded code in accordance with the at least one code generation goal," as recited in claim 1.

Claim 16 recites "graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equations," which is similar to "graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equations" as recited in claim 1. Claim 16 recites "the graphical model being capable of simulation based on the equations," which is similar to "the graphical model being capable of simulation based on the equations" as recited in claim 1. Claim 16 recites "the acquired at least one code generation goal being used to generate embedded code from the graphical model, the acquired at least one code generation goal relating to a characteristic of the code to be generated from the graphical model," which is similar to "the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment, the at least one code generation goal relating to a characteristic of the code to be generated from the graphical model" as recited in claim 1. Claim 16 recites "generating embedded code in accordance with the acquired at least one code generation goal," which is similar to "generating embedded code in accordance with the at least one code generation goal," as recited in claim 1.

Claim 30 recites "graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equations," which is similar to "graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equations" as recited in claim 1. Claim 30 recites "the graphical model being capable of simulation based on the equations," which is similar to "the graphical model being capable of simulation based on the equations" as recited in claim 1. Claim 30 recites "the acquired at least one code generation goal being used to generate embedded code from the graphical model, the acquired at least one code generation goal relating to a characteristic

of the code to be generated from the graphical model," which is similar to "the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment, the at least one code generation goal relating to a characteristic of the code to be generated from the graphical model" as recited in claim 1. Claim 30 recites "one or more instructions for generating embedded code in accordance with the acquired at least one code generation goal," which is similar to "generating embedded code in accordance with the at least one code generation goal," as recited in claim 1.

Claim 31 recites "graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equations," which is similar to "graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equations" as recited in claim 1. Claim 31 recites "the graphical model being capable of simulation based on the equations," which is similar to "the graphical model being capable of simulation based on the equations" as recited in claim 1. Claim 31 recites "the code generation goal being used to generate embedded code from the graphical model, the code generation goal relating to a characteristic of the code to be generated from the graphical model," which is similar to "the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment, the at least one code generation goal relating to a characteristic of the code to be generated from the graphical model" as recited in claim 1. Claim 31 recites "one or more instructions for generating embedded code in accordance with the code generation goal," which is similar to "generating embedded code in accordance with the at least one code generation goal," as recited in claim 1.

Claim 32 recites "graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equations," which is similar to "graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equations" as recited in claim 1. Claim 32 recites "the graphical model being capable of simulation based on the equations," which is similar to "the graphical model being capable of simulation based on the equations" as recited in claim 1. Claim 32 recites "the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment, the at least one code generation goal relating

to a characteristic of the code to be generated from the graphical model," which is similar to "the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment, the at least one code generation goal relating to a characteristic of the code to be generated from the graphical model" as recited in claim 1. Claim 30 recites "one or more instructions for generating embedded code in accordance with the at least one code generation goal," which is similar to "generating embedded code in accordance with the at least one code generation goal," as recited in claim 1.

Claim 33 recites "graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equations," which is similar to "graphical model representing a dynamic system having time-changing behavior modeled with differential, difference, and/or algebraic equations" as recited in claim 1. Claim 33 recites "the graphical model being capable of simulation based on the equations," which is similar to "the graphical model being capable of simulation based on the equations" as recited in claim 1. Claim 33 recites "a process for preparing a graphical model for a code generation process for creating code based on the graphical model and at least one code generation goal, wherein the at least one code generation goal relates to a characteristic of the code," which is similar to "the at least one code generation goal being used to generate embedded code from the graphical model in a graphical modeling environment, the at least one code generation goal relating to a characteristic of the code to be generated from the graphical model" as recited in claim 1. Claim 33 recites "wherein the computer program generates code in compliance with the at least one code generation goal," which is similar to "generating embedded code in accordance with the at least one code generation goal," as recited in claim 1.

F. Claims 15, 18-25, 27-29 and 35-39.

Dependent claims 15, 18-25, 27-29 and 35-39 depend on claims 15, 18-25, 27-29 and 35-39, and therefore are believed to be allowable for at least the same reasons as above. Therefore, Applicants respectfully request that the above rejection of claims 15, 18-25, 27-29 and 35-39 be withdrawn and that claims 15, 18-25, 27-29 and 35-39 be allowed.

CONCLUSION

All of the stated grounds of rejection have been properly traversed, accommodated, or rendered moot. Applicants therefore respectfully request that the Examiner reconsider all presently outstanding rejections and that they be withdrawn. Applicants believe that a full and complete reply has been made to the outstanding Office Action and, as such, the present application is in condition for allowance. If the Examiner believes, for any reason, that personal communication will expedite prosecution of this application, the Examiner is hereby invited to telephone the undersigned at the number provided.

Prompt and favorable consideration of this Amendment is respectfully requested.

Dated: 3/23/09

Respectfully submitted,

By [signature]

Michael A. Sartori, Ph.D.
Registration No.: 41,289
Kyle D. Petaja
Registration No.: 60,309
VENABLE LLP
P.O. Box 34385
Washington, DC 20043-9998
(202) 344-4000
(202) 344-8300 (Fax)
Attorney/Agent For Applicants

#1021421v1